Journal of Mobile Multimedia, Vol. 1, No.2 (2005) 149-160


# EFFICIENT IMAGE INDEXING AND RETRIEVAL OVER MOBILE DEVICES

SIMON SO

*Hong Kong Institute of Education*
*swwso@ied.edu.hk*

Received February 25, 2005
Revised June 17, 2005

The multimedia capabilities of mobile phones are rapidly increasing in recent years. It is now common that users can send and retrieve images over mobile phones. Image searching and retrieval is fundamentally becoming an important operation and yet there isn't any efficient and effective way to do this over mobile phones. The situation is particularly more acute when users attempt to access a content provider with a large collection of image contents. Any aimless browsing of the images will translate into air-time costs. Therefore, an efficient and effective mechanism to retrieve images over mobile phones must be sought.

In this paper, we attempt to advance this area of mobile multimedia by providing a data model as well as a query model for searching images efficiently and effectively. The data model is a semantically rich structure for representing any salient features in the image contents. XML Schema is used to model the data structure and a number of examples are illustrated in this paper. The query model works intimately with the data model and it goes beyond simple Boolean type queries. The query model supports ranked queries from the repository of image contents. The most precisely matched images will be delivered first. Users can browse the ranked images through the logical expression of the query over mobile phones. The system architecture to support the indexing and retrieval of images in our data and query model over mobile devices is described. As the popularity of micro-browsers on mobile phones capable of retrieving XTHML-MP pages over WAP gateways increase, it provides an ideal opportunity for us. We can bridge the wireless world with the Internet world seamlessly. A simulation of the concepts using XTHML-MP with WAP 2.0 is provided.

*Key words*: Mobile Multimedia, Image Retrieval on Mobile Devices, XML Schema, XTHML-MP
*Communicated by*: D Taniar

## 1 Introduction

Today, it is common that mobile phones are equipped with a higher resolution screen (e.g. 65K colors), a camera (e.g. over 1 megapixels), a larger or external storage (e.g. MultiMediaCards), a short video recorder (e.g. 10 minutes audio and video capturing) and supported streaming multimedia files (e.g. MMS SMIL). The operating system in a mobile phone is now capable of running a reasonable size multimedia application (e.g. J2ME MIDP and Symbian) and, above all, it is still called a mobile phone!

Searching and retrieving images through mobile phones are becoming one of the fundamental operations. Currently, there isn't any efficient and effective way to do this. Imagine that a mobile phone user has to search a large collection of images located at a content provider, it will not only waste a lot of time as well as air-time costs if the searching method is not efficient. In this paper, we attempt to tackle this problem. The data model and the query model of image indexing and retrieval are presented in Section 2 and Section 3 respectively. The models are particularly useful not only for the networked environment but also for the wireless world. In Section 4, a simple example to illustrate the working of the query model is provided. The system architecture to facilitate the operations through XTHML-MP over WAP 2.0 is presented in Section 5. Furthermore, the user interface consideration unique to mobile operations will also be explored. A simulation of the concepts is demonstrated. A conclusive remark is provided in Section 6.

## 2 The Data Model

To facilitate efficient image search [14], a highly structured data model is required. The Ternary Fact Model (abbr. TFM) was proposed and implemented in the past [3][13][15]. It is a canonical description of image contents to provide a set of homogenous structured data items to facilitate searching. The basic building block of the model consists of discrete facts. Conceptually, image facts may be classified into five types:

1. *Elementary Facts*: An elementary fact merely states the presence of a particular item in the image. Examples: boat, grape, tree.
2. *Modified Facts*: These are the elementary facts augmented with descriptive properties through the use of modifiers. Examples, red apple, black cat.
3. *Outline Facts*: These outline the abstraction of the images. Examples, explosion, wedding.
4. *Binary Facts*: These are the facts linking together exactly two elementary or modified facts. Such links may correspond to verbs. Example, boy rides elephant.
5. *Ternary Facts*: These are the facts linking together exactly any of three elementary, modified or outline facts together. Example, boy hits cat with racket.

Any canonical description of image contents can then be represented and indexed by the combination of the facts. The XML Schema [9][17][18][19][20] for Ternary Fact Model is defined as follows:

```
<xsd:complexType name="TernaryFactModelType">
  <xsd:choice>
    <xsd:sequence>
      <xsd:element name="Fact">
        <xsd:complexType>
          <xsd:simpleContent>
            <xsd:extension base="xsd:token">
              <xsd:attribute name="modifier" type="xsd:token"/>
            </xsd:extension>
          </xsd:simpleContent>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
    <xsd:sequence>
      <xsd:element name="Link" type="xsd:token"/>
      <xsd:element name="Fact" minOccurs="2" maxOccurs="3">
        <xsd:complexType>
```

```
            <xsd:simpleContent>
              <xsd:extension base="xsd:token">
                <xsd:attribute name="modifier" type="xsd:token"/>
              </xsd:extension>
            </xsd:simpleContent>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
  </xsd:choice>
</xsd:complexType>
```

The complex type, TernaryFactModelType, is a selection of two choices. The first choice covers the Elementary Facts, Modified Facts, and Outline Facts. The second choice covers the Binary Facts and Ternary Facts. The minOccurs and maxOccurs for Fact in the second choice are set at 2 and 3 respectively to meet the cardinalities for Binary fact and Ternary Fact. Furthermore, modifier is an optional attribute for Fact element. An element of TernaryFactModelType can then be defined as:

…

```
<xsd:element name="TernaryFactModel" type="TernaryFactModelType"/>
```

…

Examples for *Elementary Fact*, *Outline Fact*, *Modified Fact*, *Binary Fact* and *Ternary Fact* would be:

1. *Elementary Fact*:
```
<TernaryFactModel>
    <Fact>boat</Fact>
</TernaryFactModel>
```

2. *Modified Fact*:
```
<TernaryFactModel>
    <Fact modifier="red">car</Fact>
</TernaryFactModel>
```

3. *Outline Fact*:
```
<TernaryFactModel>
    <Fact>explosion</Fact>
</TernaryFactModel>
```

4. *Binary Fact*:
```
<TernaryFactModel>
    <Link>ride</Link>
    <Fact>boy</Fact>
    <Fact modifier="white">elephant</Fact>
</TernaryFactModel>
```

5. *Ternary Fact*:
```
<TernaryFactModel>
    <Link>hit</Link>
    <Fact modifier="young">boy</Fact>
    <Fact modifier="red">car</Fact>
    <Fact>racket</Fact>
</TernaryFactModel>
```

## 3. The Query Model

For any information retrieval, a query model is an important component next to the data model [10][12]. Generally, there are two broad categories of queries in image retrieval. They are Boolean queries and ranked (similarity) queries. A Boolean query should, in fact, be a special case of a ranked query. In other words, if the image set resulted from a Boolean query is not empty, it should be ranked at the top for any similarity query to be reasonable and meaningful. Hence, the query model for our data model is based on ranking.

Similar to the conventional query processing, the steps involved in processing an image query include: (1) parsing and validating a query, (2) translating and transforming into an intermediate query, (3) optimizing the intermediate query, (4) formulating the execution plan, and lastly (5) performing the query. Figure 1 shows the processing steps with an illustrative example.

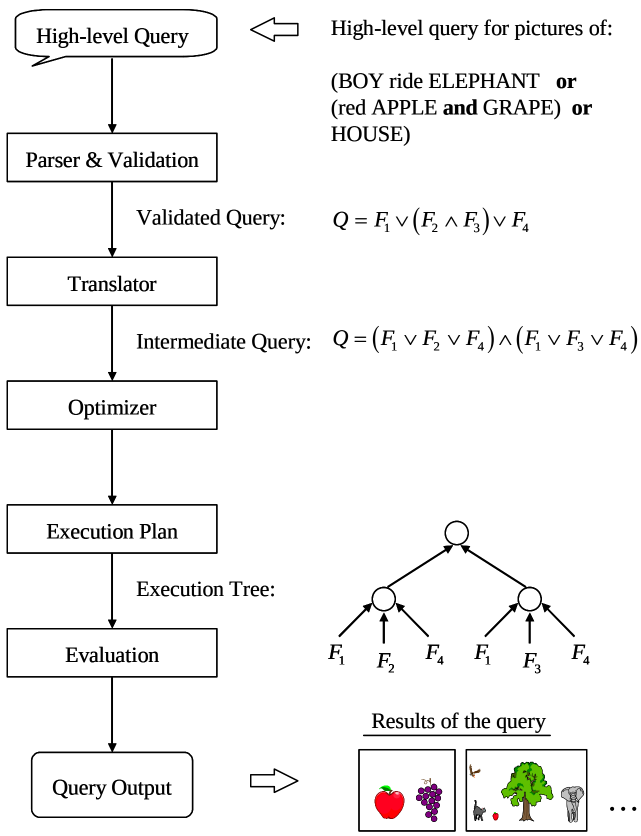


Figure 1. The processing steps for the query model with an illustration

Formally, let $F_i$ be either a TFM fact in a general expression of a query after the parsing and validating processes and presented in the form of an Boolean equation, $Q(F_1, F_2, ..., F_n)$. For example, $Q = ((F_1 \wedge F_2) \vee (F_1 \wedge F_3) \vee F_4)$ for $n$ = 4. Allowable Boolean operators include *and* $(\wedge)$, *or* $(\vee)$ , *not* $(\neg)$ [a] together with parentheses. The general expression is required to undergo translation and

[a] For simplicity, this logical operator is left out for later implementation.

transformation into an intermediate query. It is impossible to evaluate the query with arbitrary levels of parentheses and terms. The intermediate form that we use is a *conjunction of disjunctions*. That is, any general Boolean equation, is translated into the following form,

$$Q(F_1, F_2, \ldots, F_n) = (F_w \vee \ldots \vee F_x) \wedge \ldots \wedge (F_y \vee \ldots \vee F_z)$$

where $w < x$ and $y < z$ such that $w, x, y, z \in \{1 \ldots n\}$. The reasons to use this form instead of a *disjunction of conjunctions* are: (1) the logical integrity is maintained for the disjunctive terms when we relax in the ranking process as described later, (2) we can perform lazy evaluation when the query is performed, and (3) a long list of disjunctive expressions are rarely occurred in image retrieval.

The process of optimization is an important step. For image retrieval, the total number of TFM facts is expected to be manageable for a smaller repository. Therefore, we will leave this important research topic for another research initiative.

Once the order of execution for each disjunction is set, the execution plan can be formulated. An execution tree for our configuration similar to Figure 2 is used to propagate the intermediate result. For a Boolean query, this is the final result and the query stops. For a ranked query, the process continues and the heuristics described next does it without resorted to weighting.

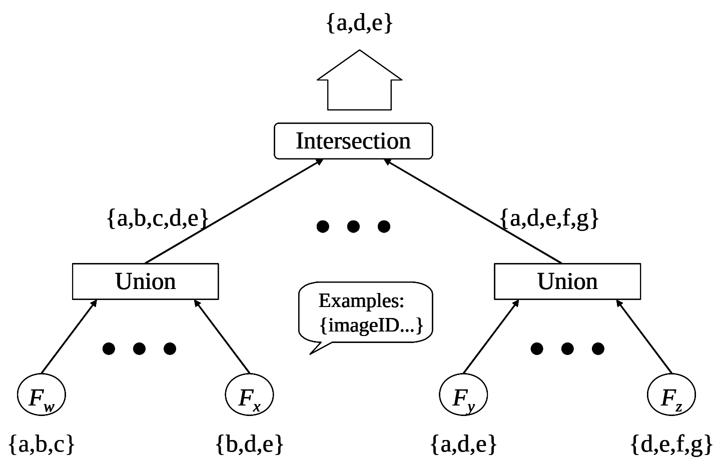


Figure 2. An execution tree for target images

The selection criteria for our ranked queries should be natural enough so that the results are consistent and meet our expectation. For each iteration, the criteria should be relaxed to accommodate less precise answers. Figure 3 depicts the heuristics.

The criteria are briefly explained as follows:

1. *Boolean evaluation*: This step provides the precise solution with a given query expression. TFM facts are matched completely.
2. *Substitution*: This optional step is to replace the facts using additional tools such as thesauri or knowledge-based tools.
3. *Dropping links*: For any binary or ternary fact, the link is removed. This will give rise to two or three individual elementary, outline or modified facts. The situation is nicely fitted under the disjunctive relationship.

4. *Dropping modifiers*: To relax further, the modifiers for all modified facts are deleted and become elementary, outline or modified facts.
5. *Dropping elementary facts*: Individual facts are dropped to relax the disjunctive nature of the query.

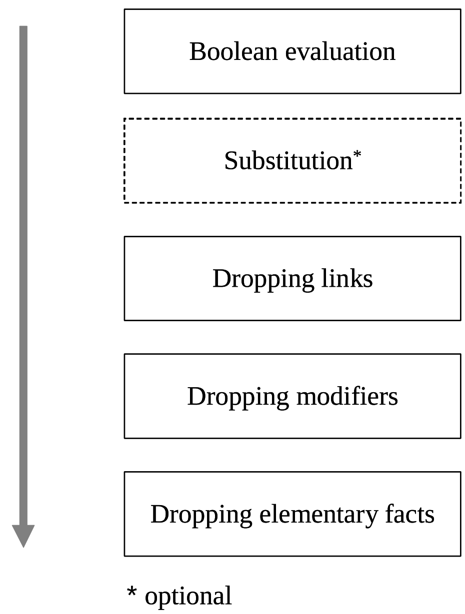


Figure 3. The heuristics for selecting images

Each pass of the execution tree will produce a set of incremental candidates to be inserted into the final target set of images as shown in Figure 4. The set is somewhat ordered with the most likely candidates at the top. It is obvious that the flexibility of stopping at any one of the four selection steps is highly possible.

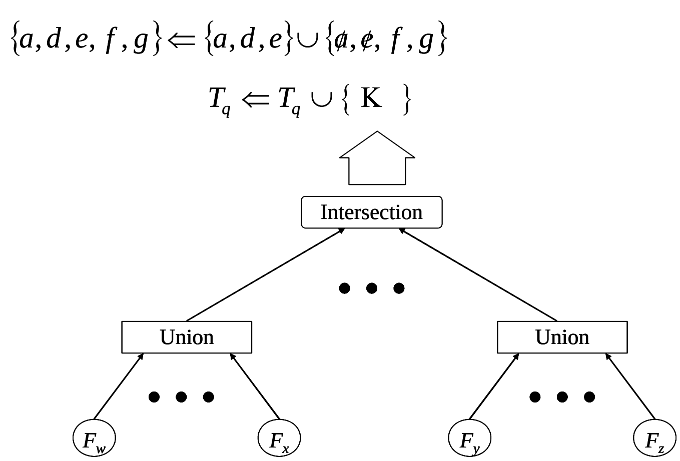


Figure 4. The set of target images

### 4. A Query Example for the Algorithm

To understand the heuristics of the algorithm, a simple example is used to illustrate the concepts and the effectiveness of our algorithm to undergo a ranked query processing. Let us assume we have three sample images as shown in Figure 5. The images are properly indexed using the data model as described in Section 2. The apples in Image A and C are red in colour.

(A) (B) (C)

Figure 5. Three Sample Images

Let us also assume we want to locate images with the following query,

(BOY Ride ELEPHANT or (GRAPE and green APPLE) or DOG or HOUSE)

The query is parsed into the following TFM terms,

| | |
|---|---|
| $F_1$ = BOY Ride ELEPHANT | (a binary fact) |
| $F_2$ = GRAPE | (an elementary fact) |
| $F_3$ = green APPLE | (a modified fact) |
| $F_4$ = DOG | (an elementary fact) |
| $F_5$ = HOUSE | (an elementary fact) |

The parsed query becomes,

$$Q(F_1, F_2, F_3, F_4, F_5) = (F_1 \vee (F_2 \wedge F_3) \vee F_4 \vee F_5)$$

The intermediate form of the *conjunction of disjunctions* becomes,

$$Q(F_1, F_2, F_3, F_4, F_5) = (F_1 \vee F_2 \vee F_4 \vee F_5) \wedge (F_1 \vee F_3 \vee F_4 \vee F_5)$$

Let us also assume $Q_1 = (F_1 \vee F_2 \vee F_4 \vee F_5)$ and $Q_2 = (F_1 \vee F_3 \vee F_4 \vee F_5)$ are the two disjunctive terms in $Q$. The resulting ranked query set is empty, $T_q = \varnothing$, initially. The following table summarizes the processes of our algorithm with the ranked images {B,A,C} as the result. It matches our expectation for the logical sequence of images to be delivered with the given query.

| | Boolean evaluation | Dropping links | Dropping modifiers | Dropping elementary facts |
|---|---|---|---|---|
| $F_1$ | ∅ | {A} Note #1 | {A} | N/A |
| $F_2$ | {C} | {C} | {C} | N/A |
| $F_3$ | ∅ | ∅ | {A,C} Note #2 | N/A |
| $F_4$ | ∅ | ∅ | ∅ | ∅ |
| $F_5$ | {B} | {B} | {B} | N/A |
| $Q_1$ | {B,C} | {A,B,C} | {A,B,C} | ∅ |
| $Q_2$ | {B} | {A,B} | {A,B,C} | ∅ |
| $Q$ | {B} | {A,B} | {A,B,C} | ∅ |
| $T_q$ | {B} | {B,A} | {B,A,C} | {B,A,C} |

Note #1: The link, "Ride", is dropped. ELEPHANT is matched.

Note #2: The modifier, "green", is dropped. APPLE is matched on red APPLE.

Table 1. The query processing (from left to right)

## 5. The System Architecture and User Interaction

In contrast with PCs, most mobile devices have slower processors, limited memory, and smaller form factors. Hence, we need to minimize the processing in the client devices [4][6]. Any heavy processing should be offloaded to the server side [11]. A WAP gateway connected to a Web server for our application is desirable [1][5]. Figure 6 outlines the system architecture to support efficient image indexing and retrieval of our data and query model over mobile devices.

A micro browser to support searching and browsing was tested [7][8]. XHTML-MP over WAP 2.0 is used for our simulation [2][16]. The reason to choose this is that a familiar look and feel to the fixed Internet world could likely ease the intimidation when the user interacts with the phone in searching images. The user interface must also address the unique characteristics of the mobile users interacting with phones. For examples, the data entry or browsing will be done without a mouse. The interaction will often be one finger only. The screen is usually small and difficult in navigation. Inputting text characters with a phone keypad will be a frustrating experience for the novice user. In designing the user interface, we would like to keep the navigation simple. Also, the text input is kept to the minimum. Figure 7 demonstrates our user interface design. We deliberately avoid the input of a general expression for a query. Instead, a two-level logical composition is provided. *Any* or *all* of the TFM facts can be built on each level by following a simple syntax. We believe that this design is sufficient for various kinds of queries.

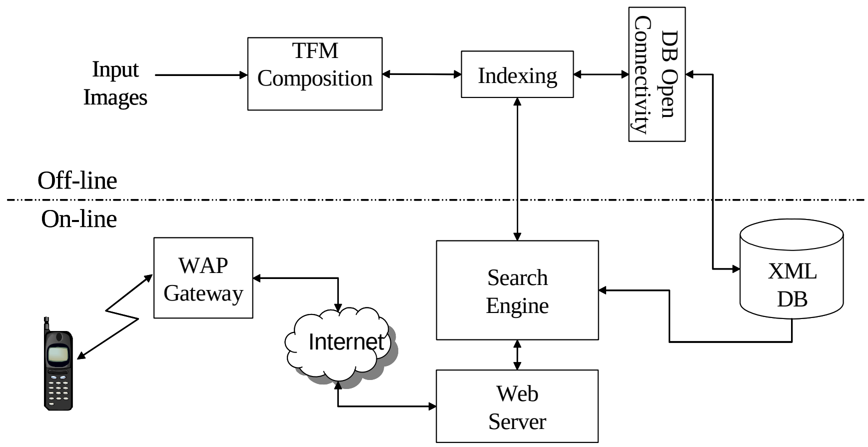


Figure 6. The system architecture to support image indexing and retrieval over mobile devices

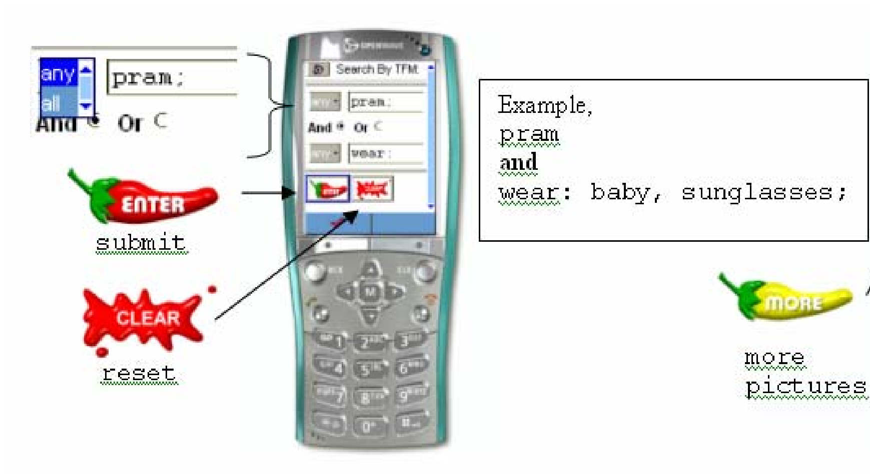



Figure 7. The user interface design

Figure 8. The result

To enter multiple TFM facts on one line, the user needs to separate each TFM fact by a semicolon. A colon is used to separate <Link> and a comma for each <Fact>. At least a space for modifier is also required. For example, "bottle; crawl: happy baby, deck;" is a syntactically correct input.

Figure 8 demonstrates a precise query of `Q = pram and (wear: baby, sunglasses)` from Figure 7. An example to illustrate a ranked query is shown in Figure 9.

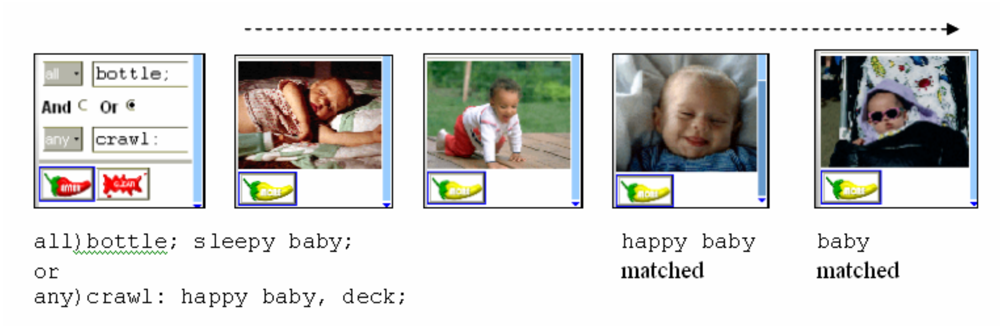


Figure 9. A ranked query

To provide the reader a better understanding of the implementation in XHTML-MP, the codes to generate the user interface of Figure 7 is shown below.

```
<?xml version="1.0" charset="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN"
"http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
<html>
<head>
<link rel="stylesheet" href="style.css" type="text/css"/>
<title>
Search By TFM:
<title>
</head>
<body>
<form method="GET" action="retrieval.asp">
   <hr height="3"/>
      <table border="0">
         <tr>
            <td width="15%">
               <select name="op1">
                  <option>any</option>
                  <option>all</option>
               </select>
            </td>
            <td width="30%">
               <input name="Line1" type="text" size="12"
                maxlength="80" height="1"/>
            </td>
         </tr>
      </table>
      <table border="0">
         <tr>
            <td>
               <label><b> And </b><input type="radio" name="logic1"
                value="And1" checked="checked"/></label>
            </td>
            <td>
               <label><b> Or </b><input type="radio" name="logic1"
```

```
                value="Or1" /></label>
            </td>
         </tr>
      </table>
      <table border="0">
         <tr>
            <td width="15%">
               <select name="op2">
                  <option>any</option>
                  <option>all</option>
               </select>
            </td>
            <td width="30%">
               <input name="Line2" type="text" size="12"
                maxlength="80" height="1"/>
            </td>
         </tr>
      </table>
   <hr height="3"/>
   <input type="submit" src="enter_red.gif" width="35" height="20"
    value="Retrieve"/>
   <input type="reset" src="clear.gif" width="35" height="20"
    value="Clear"/>
</form>
</body>
</html>
```

## 6. Conclusion

In this paper, the data model together with the query model for image indexing and retrieval over mobile devices is presented. We demonstrated that a carefully structured canonical description could be an efficient and effective model for searching images through a wireless network with limited bandwidth. Our approach can avoid the mobile user to browse aimlessly over an image repository and subsequently reduce air-time costs. Also, by making use of the latest WAP 2.0 standard, it provides the novice users the benefits of both wireless and fixed worlds. The users can browse with the usual look and feel as if they are on the Internet. In designing our user interface, the unique characteristics of handheld interaction are being considered for the purposes of reducing the frustration of mouse-less navigation and the small form factor associated with most mobile devices. We understand there are many ways of refining our data model and creating better heuristics for our query model. Nevertheless, this research provides an adequate approach for searching images over mobile devices.

## References


1. Graves, M., *Designing XML Databases*, Upper Saddle River NJ: Prentice Hall PTR, 2002
2. Hillborg, M., *Wireless XML Developer's Guide*, Berkeley CA: Osborne/McGraw-Hill, 2002
3. Leung, C.H.C., and Zheng, Z.J., "Image Data Modelling for Efficient Content Indexing," *Proc. International Workshop on Multimedia Database Management Systems*, New York, August 1995, IEEE Computer Society Press, pp.143-150
4. Lin, Y.B, and Chlamtac, I., *Wireless and Mobile Network Architectures*, New York: Wilsey, 2001

5. Linsenbardt, M.A., and Stigler, N.S., *SQL Server 2000 Administration*, Berkeley CA: Osborne/McGraw-Hill, 2001
6. Mallick, M., *Mobile and Wireless Design Essentials*, New York: Wilsey 2003
7. Nokia Forum http://www.forum.nokia.com
8. Openwave http://www.openwave.com
9. Sall, K.B., *XML Family of Specifications: A Practical Guide*, Boston MA: Pearson Education, 2002
10. Salton, G., *Automatic Text Processing*, Reading MA: Addison-Wesley, 1989
11. Shapiro, J.R., *SQL Server 2000: the Complete Reference*, Berkeley CA: Osborne/McGraw Hill, 2001
12. Witten I., Moffat A., and Bell T., *Managing Gigabytes: Compressing and Indexing Documents and Images*, 2nd Ed., San Francisco CA: Morgan Kaufmann Publishers, 1999
13. So, W.W.S. , and Leung, C.H.C., "Inverted Image Indexing and Compression," *SPIE Proc. Multimedia Storage and Archiving Systems II*, Dallas, Texas, Vol. 3229, 3-4 November 1997, pp.254-263
14. So, S. and Leung C., "A New Paradigm in Image Indexing and Retrieval using Composite Bitplane Signatures," *IEEE International Conference on Multimedia Computing and Systems*, Florence, Vol. I, 7-11 June 1999, pp.855-859
15. So, S., Leung, C., and Tse, P., "An XML Data Model for Inverted Image Indexing," *SPIE Proc. Internet Imaging IV*, Santa Clara, CA, Vol. 5018, 20-24 January 2003, pp.190-197
16. WAP 2.0 Standard http://www.wapforum.org
17. Wyke, R.A., Rehman, S., and Leupen, B., *XML Programming (Core Reference)*, Redmond, WA: Microsoft Press, 2002
18. XML Schema Part 0: Primer http://www.w3.org/TR/xmlschema-0/
19. XML Schema Part 1: Structure http://www.w3.org/TR/xmlschema-1/
20. XML Schema Part 2: Datatypes http://www.w3.org/TR/xmlschema-2/